REMARKS

Claims 1-25 are pending in the present application.

Claims 1-25 have been rejected.

Claims 1-25 remain in the present application. Reconsideration of the claims in view of the following arguments is respectfully requested.

In Sections 6-32 of the October 24, 2003 Office Action, the Examiner rejected Claims 1-25 under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,630,066 to *Gosling* (hereafter, "*Gosling*") in view of Jones et al., "The Spineless Tagless G-machine" (hereafter, "*Jones*"). The Applicant respectfully traverses the rejection of Claims 1-25.

The Applicant directs the Examiner's attention to Claim 1, which contains following unique and novel limitations:

> 1. (Original) A method for detecting corruption associated with a stack in a storage device, the method comprising the steps of:
> inserting a quantity of information adjacent to the stack in the storage device, the quantity of information having an initial state; and
> inspecting the quantity of information so as to identify any deviation from the initial state and thereby detect corruption associated with the stack in the storage device. (emphasis added).

The Applicant respectfully asserts that the above-emphasized limitations are not disclosed, suggested, or even hinted at in the *Gosling* reference or the *Jones* reference, or in the combination of the *Gosling* and *Jones* references.

In rejecting Claim 1, the Examiner asserted that the Claim 1 limitation regarding "inspecting the quantity of information so as to identify any deviation . . ." is disclosed in the *Gosling* reference

at column 10, lines 34-46, and at column 12, lines 43-59. The Examiner also asserted that the Claim 1 limitation regarding "inserting a quantity of information adjacent the stack . . ." is disclosed in the *Jones* reference at page 195, column 1, lines 16-23.

The Applicant has reviewed the portions of the *Gosling* reference and the *Jones* reference relied upon by the Examiner and respectfully asserts that the Examiner has misunderstood the teachings of the *Gosling* reference and the *Jones* reference. The text of the *Gosling* reference relied upon by the Examiner at column 10, lines 34-46 states:

> In operation, the verifier 240 processes each bytecode instruction which requests datum to be popped off the stack and pops off the same number of data type values off the virtual stack 344. The verifier then compares the "popped" data type values from the virtual stack 344 with the data type requirements of the bytecode instruction. Similarly, for each bytecode instruction requesting datum to be pushed onto the stack, the verifier pushes onto the virtual stack a corresponding data type value.
>
> One aspect of program verification in accordance with present invention is verification that the number and data type of the operands in the operand stack status is identical every time a particular instruction is executed. If a particular bytecode instruction can be immediately preceded in execution by two or more different instructions, then the virtual stack status immediately after processing of each of those different instructions must be compared. Usually, at least one of the different preceding instructions will be a conditional or unconditional jump or branch instruction. A corollary of the above "stack consistency" requirement is that each program loop must not result in a net addition or reduction in the number of operands stored in the operand stack.

Additionally, the text of the *Gosling* reference relied upon by the Examiner at column 12, lines 43-59 states:

> If a stack snapshot has been stored for the currently selected instruction (indicating that a jump instruction associated with this target instruction has already been processed), then the verifier compares (438) the virtual stack snapshot information stored in the snapshot portion 350 of the stack snapshot storage structure

> 346 for the currently selected instruction with the current state of the virtual stack. If the comparison shows that the current state and the snapshot do not match, then an error message or signal is generated (440) identifying the place in the bytecode program where the stack status mismatch occurred. In the preferred embodiment, a mismatch will arise if the current virtual stack and snapshot do not contain the same number or types of entries. The verifier will then set a verification status value 245 for the program to false, and abort (442) the verification process. Setting the verification status value 245 for the program to false prevents execution of the program by the bytecode interpreter 242 (FIG. 3).

The Applicant respectfully asserts that the portions of the *Gosling* reference relied upon by the Examiner do not disclose, suggest, or even hint at the unique and novel limitations recited above in Claim 1. The portions of the *Gosling* reference relied upon by the Examiner never discuss inspecting the contents of a memory location adjacent to the stack space. Rather, the device disclosed in the *Gosling* reference discloses that the verifier 240 compares the contents inside the stack with the contents of a virtual stack to detect mismatches. Moreover, the verifier 240 does not compare the contents of the stack to a fixed value (i.e., an initial state). Instead, the *Gosling* device compares the changing value in the stack with the changing value in the virtual stack to determine that they are always the same, even though both may change together. In short, the *Gosling* reference proposes a different solution to a different problem than does the unique and novel method recited in Claim 1.

Furthermore, the *Jones* reference does nothing to overcome the shortcomings of the *Gosling* reference. The text of the *Jones* reference relied upon by the Examiner at page 195, column 1, lines 16-23, states:

> Given a single stack, there needs to be some way of distinguishing pointers from non-pointers. This can be done by tagging each stack element with a bit to say whether it is a pointer or not, either by stealing one of the 32 bits in a machine word, or by additionally stacking the tag bit. We believe that this approach imposes an

> unacceptable overhead on stack operations, and we have developed an alternative technique.

The Applicant respectfully asserts that the portions of the *Jones* reference relied upon by the Examiner do not disclose, suggest, or even hint at the unique and novel limitations recited above in Claim 1. The cited portions of the *Jones* reference never mention inspecting the contents of a memory location adjacent to the stack space and never mention comparing those contents to a fixed value (i.e., an initial state).

In sum, the unique and novel limitations recited in Claim 1 are not disclosed, suggested, or even hinted at in the *Gosling* reference or the *Jones* reference, or in the combination of the *Gosling* and *Jones* references. This being the case, Claim 1 presents patentable subject matter over the *Gosling* reference and the *Jones* reference. Also, Claims 2-10 depend from Claim 1 and contain all of the unique and novel limitations recited in Claim 1. Therefore, Claims 2-10 also are patentable over the *Gosling* and *Jones* references.

The Applicant respectfully asserts that independent Claims 11 and 21 contain limitations that are analogous to the unique and novel limitations recited in Claim 1. This being the case, independent Claims 11 and 21 present patentable subject matter over the *Gosling* reference and the *Jones* reference. Finally, Claims 12-20, which depend from Claim 11, and Claims 22-25, which depend from Claim 21, contain all of the unique and novel limitations recited in Claims 11 and 21, respectively. Therefore, Claims 12-20 and 22-25 are patentable over the *Gosling* and *Jones* references.

SUMMARY

For the reasons given above, the Applicant respectfully requests reconsideration and allowance of pending claims and that this Application be passed to issue. If any outstanding issues remain, or if the Examiner has any further suggestions for expediting allowance of this Application, the Applicant respectfully invites the Examiner to contact the undersigned at the telephone number indicated below or at ***jmockler@davismunck.com***.

The Commissioner is hereby authorized to charge any additional fees connected with this communication or credit any overpayment to Deposit Account No. 50-0208.

Respectfully submitted,
DAVIS MUNCK, P.C.

Date: 23 Dec. 2003

John T. Mockler
Registration No. 39,775

P.O. Drawer 800889
Dallas, Texas 75380
Phone: (972) 628-3600
Fax: (972) 628-3616
E-mail: *jmockler@davismunck.com*